ATSC Standard:
ATSC 3.0 Security and Service Protection

Doc. A/360:2022-03
31 March 2022

Advanced Television Systems Committee
1300 I Street, N.W., Suite 400E
Washington, D.C. 20005
202-872-9160

The Advanced Television Systems Committee, Inc. is an international, non-profit organization developing voluntary standards and recommended practices for broadcast television and multimedia data distribution. ATSC member organizations represent the broadcast, professional equipment, motion picture, consumer electronics, computer, cable, satellite, and semiconductor industries. ATSC also develops implementation strategies and supports educational activities on ATSC standards. ATSC was formed in 1983 by the member organizations of the Joint Committee on Inter-society Coordination (JCIC): the Consumer Technology Association (CTA), the Institute of Electrical and Electronics Engineers (IEEE), the National Association of Broadcasters (NAB), the Internet & Television Association (NCTA), and the Society of Motion Picture and Television Engineers (SMPTE). For more information visit www.atsc.org.

Note: The user's attention is called to the possibility that compliance with this standard may require use of an invention covered by patent rights. By publication of this standard, no position is taken with respect to the validity of this claim or of any patent rights in connection therewith. One or more patent holders have, however, filed a statement regarding the terms on which such patent holder(s) may be willing to grant a license under these rights to individuals or entities desiring to obtain such a license. Details may be obtained from the ATSC Secretary and the patent holder.

Implementers with feedback, comments, or potential bug reports relating to this document may contact ATSC at https://www.atsc.org/feedback/.

Revision History

Version	Date
Candidate Standard approved	28 October 2016
Updated CS approved	27 March 2017
A/360:2018 Standard approved	9 January 2018
Candidate Standard Revision approved	29 August 2018
Updated CS approved	23 January 2019
A/360:2019 Standard approved	20 August 2019
Capitalization errors in Table A.2 “Prefix” column corrected	24 January 2020
A/360:2019 Amendment No. 1 approved	22 May 2020
A/360:2019 Amendment No. 2 approved	16 February 2021
A/360:2019 Amendment No. 3 approved	6 January 2022
A/360:2019 Amendment No. 3 approved	10 January 2022
A/360:2022-01 published (a rollup of Amendments 1 – 4 and reference updates)	10 January 2022
A/360:2022-03 published (references to ATSC documents updated)	31 March 2022

Table of Contents

1. SCOPE 1
   1.1 Organization 1
2. REFERENCES 1
   2.1 Normative References 1
   2.2 Informative References 3
3. DEFINITION OF TERMS 3
   3.1 Compliance Notation 3
   3.2 Treatment of Syntactic Elements 3
      3.2.1 Reserved Elements 3
   3.3 Acronyms and Abbreviations 4
   3.4 Terms 5
   3.5 Extensibility 6
   3.6 XML Schema and Namespace 6
4. SYSTEM OVERVIEW 7
   4.1 Features 7
   4.2 System Architecture 7
   4.3 Central Concepts 7
5. SPECIFICATION 8
   5.1 Transport Protection 8
      5.1.1 Internet Streaming Transport Security 8
   5.2 ATSC 3.0 Cryptographic Signing 11
      5.2.1 ATSC 3.0 Application Code Signing 12
      5.2.2 ATSC 3.0 Signaling Message Signing 12
   5.3 Certificates and Certificate Management 17
      5.3.1 Certificate Profiles 18
   5.4 ATSC 3.0 Client Certificate Storage 19
   5.5 Certificate Revocation and Status Information 19
      5.5.1 Certificate Revocation and Status Information for TLS Server Certificates 20
      5.5.2 Certificate Revocation and Status Information for ATSC 3.0 Application Signing Certificates 20
   5.6 Pre-Shared Key Encrypted Connections 20
      5.6.1 Pre-Shared Key Registration 21
      5.6.2 TLS 1.3 Pre-Shared Key Exchange Parameters 22
   5.7 Content Protection 22
      5.7.1 Common Encryption 22
      5.7.2 CENC and EME Support 22

ANNEX A: ASN .1 OBJECT IDENTIFIERS 23
   A.1 ATSC Registered Object Identifiers 23
   A.2 Other Referenced Object Identifiers 23

Index of Tables

Table 5.1 CertificationData XML Format 14
Table 5.2 CMS Signed Data XML Format 17
Table A.1 ATSC Registered Object Identifiers 23
Table A.2 Other Referenced Object Identifiers 23

ATSC Standard:
ATSC 3.0 Security and Service Protection

1. SCOPE

This standard specifies the mechanisms for security and service protections in ATSC 3.0 systems.

1.1 Organization

This document is organized as follows:

- Section 1 – Outlines the scope of this document and provides a general introduction.
- Section 2 – Lists references and applicable documents.
- Section 3 – Provides a definition of terms, acronyms, and abbreviations for this document.
- Section 4 – System overview
- Section 5 – Specification
- Annex A: – ROUTE/DASH Client Processing for CENC and EME

2. REFERENCES

All referenced documents are subject to revision. Users of this Standard are cautioned that newer editions might or might not be compatible.

2.1 Normative References

The following documents, in whole or in part, as referenced in this document, contain specific provisions that are to be followed strictly in order to implement a provision of this Standard.

[1] IEEE: "Use of the International Systems of Units (SI): The Modern Metric System," Doc. SI 10, Institute of Electrical and Electronics Engineers, New York, NY.

[2] ISO/IEC: ISO/IEC 23001-7:2016, "Information technology — MPEG systems technologies — Part 17: Common encryption in ISO base media file format files."

[3] DASH: "Guidelines for Implementation: DASH-IF Interoperability Points for ATSC 3.0", Version 1.1, DASH Industry Forum, Beaverton, OR, 12 June 2018.

[4] IETF: "RFC 3279, Algorithms and Identifiers for the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile," L. Bassham, W. Polk, R. Housley, Internet Engineering Task Force, Fremont, CA, April 2002.

[5] IETF: "RFC 4033, DNS Security Introduction and Requirements," Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, Internet Engineering Task Force, Fremont, CA, March 2005.

[6] IETF: "RFC 4055, Additional Algorithms and Identifiers for RSA Cryptography for use in the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile," J. Schaad, B. Kaliski, R. Housley, Internet Engineering Task Force, Fremont, CA, June 2005.

[7] IETF: "RFC 5019, The Lightweight Online Certificate Status Protocol (OCSP) Profile for High-Volume Environments," A. Deacon, R. Hurst, Internet Engineering Task Force, Fremont, CA, September 2007.

[8] IETF: "RFC 5077, Transport Layer Security (TLS) Session Resumption without Server-Side State," J. Salowey, H. Zhou, P. Eronen, H. Tschofenig, Internet Engineering Task Force, Fremont, CA, January 2008.

[9] IETF: "RFC 5246, The Transport Layer Security (TLS) Protocol Version 1.2," T. Dierks, E. Rescorla, Internet Engineering Task Force, Fremont, CA, August 2008.

[10] IETF: "RFC 5280, Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile," D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, W. Polk, Internet Engineering Task Force, Fremont, CA, May 2008.

[11] IETF: "RFC 5289, TLS Elliptic Curve Cipher Suites with SHA-256/384 and AES Galois Counter Mode (GCM)," E. Rescorla, Internet Engineering Task Force, Fremont, CA, August 2008.

[12] IETF: "RFC 5480, Elliptic Curve Cryptography Subject Public Key Information," S. Turner, D. Brown, K. Yiu, R. Housley, T. Polk, Internet Engineering Task Force, Fremont, CA, March 2009.

[13] IETF: "RFC 5652, Cryptographic Message Syntax (CMS)," R. Housley, Internet Engineering Task Force, Fremont, CA, September 2009.

[14] IETF: "RFC 5746, Transport Layer Security (TLS) Renegotiation Indication Extension," E. Rescorla, M. Ray, S. Dispensa, N. Oskov, Internet Engineering Task Force, Fremont, CA, February 2010.

[15] IETF: "RFC 5751, Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.Message Specification," B. Ramsdell, S. Turner, Internet Engineering Task Force, Fremont, CA, January 2010.

[16] IETF: "RFC 5753, Use of Elliptic Curve Cryptography (ECC) Algorithms in Cryptographic Message Syntax (CMS)," S. Turner, D. Brown, Internet Engineering Task Force, Fremont, CA, January 2010.

[17] IETF: "RFC 5758, Internet X.509 Public Key Infrastructure: Additional Algorithms and Identifiers for DSA and ECDSA," Q. Dang, S. Santesson, K. Moriarty, D. Brown, T. Polk, Internet Engineering Task Force, Fremont, CA, January 2010.

[18] IETF: "RFC 5940, Additional Cryptographic Message Syntax (CMS) Revocation Information Choices," S. Turner, R. Housley, Internet Engineering Task Force, Fremont, CA, August 2010.

[19] IETF: "RFC 6066, Transport Layer Security (TLS) Extensions: Extension Definitions," D. Eastlake 3rd, Internet Engineering Task Force, Fremont, CA, January 2011.

[20] IETF: "RFC 6840, Clarifications and Implementation Notes for DNS Security (DNSSEC)", S. Weiler, and D. Blacka, Internet Engineering Task Force, Fremont, CA, February 2013.

[21] IETF: "RFC 6960, X.509 Internet Public Key Infrastructure Online Certificate Status Protocol – OCSP," S. Santesson, M. Myers, R. Ankney, A. Malpani, S. Galperin, C. Adams, Internet Engineering Task Force, Fremont, CA, June 2013.

[22] IETF: "RFC 8018, PKCS #5: Password-Based Cryptography Specification, Version 2.1," K. Moriarty, B. Kaliski, A. Rusch, Internet Engineering Task Force, Fremont, CA, January 2017.

[23] IETF: "RFC 8446, TLS 1.3, The Transport Layer Security (TLS) Protocol Version 1.3," Internet Engineering Task Force, Fremont, CA, [July 2018].

[24] IETF: "RFC 7539, ChaCha20 and Poly1305 for IETF Protocols," Y. Nir, A. Langley, Internet Engineering Task Force, Fremont, CA, May 2015.

[25] ITU-T: "Information technology – Open Systems Interconnection – Procedures for the operation of OSI Registration Authorities: Generation and registration of Universally Unique

Identifiers (UUIDs) and their use as ASN.1 object identifier components", Rec. X.667, International Telecommunication Union, September 2004.

[26] ATSC: "ATSC Standard: Signaling, Delivery, Synchronization, and Error Protection," Doc. A/331:2022-03, Advanced Television System Committee, Washington, DC, 31 March 2022.

[27] W3C: "XML Schema Part 2: Datatypes Second Edition" W3C Recommendation, Worldwide Web Consortium, 28 October 2004. https://www.w3.org/TR/xmlschema-2/

2.2 Informative References

The following documents contain information that may be helpful in applying this Standard.

[28] CTA: "CTA 2053. Receiver Specifications for ATSC 2.0 Security," ANSI/CTA-2053, Consumer Technology Association, Arlington, VA, August 2015.

[29] ATSC: "ATSC Standard: Companion Device," Doc. A/338:2022-03, Advanced Television System Committee, Washington, DC, 31 March 2022.

[30] CA/Browser Forum: "Baseline Requirements for the Issuance and Management of Publicly-Trusted Certificates," Version 1.5.9, CA/Browser Forum, 14 June 2018. http://cabforum.org/baseline-requirements-documents/

3. DEFINITION OF TERMS

With respect to definition of terms, abbreviations, and units, the practice of the Institute of Electrical and Electronics Engineers (IEEE) as outlined in the Institute's published standards [1] shall be used. Where an abbreviation is not covered by IEEE practice or industry practice differs from IEEE practice, the abbreviation in question will be described in Section 3.3 of this document.

3.1 Compliance Notation

This section defines compliance terms for use by this document:

shall – This word indicates specific provisions that are to be followed strictly (no deviation is permitted).

shall not – This phrase indicates specific provisions that are absolutely prohibited.

should – This word indicates that a certain course of action is preferred but not necessarily required.

should not – This phrase means a certain possibility or course of action is undesirable but not prohibited.

3.2 Treatment of Syntactic Elements

This document contains symbolic references to syntactic elements used in the audio, video, and transport coding subsystems. These references are typographically distinguished by the use of a different font (e.g., restricted), may contain the underscore character (e.g., sequence_end_code) and may consist of character strings that are not English words (e.g., dynrng).

3.2.1 Reserved Elements

One or more reserved bits, symbols, fields, or ranges of values (i.e., elements) may be present in this document. These are used primarily to enable adding new values to a syntactical structure without altering its syntax or causing a problem with backwards compatibility, but they also can be used for other reasons.

The ATSC default value for reserved bits is '1'. There is no default value for other reserved elements. Use of reserved elements except as defined in ATSC Standards or by an industry standards-setting body is not permitted. See individual element semantics for mandatory settings and any additional use constraints. As currently-reserved elements may be assigned values and meanings in future versions of this Standard, receiving devices built to this version are expected to ignore all values appearing in currently-reserved elements to avoid possible future failure to function as intended.

3.3 Acronyms and Abbreviations

The following acronyms and abbreviations are used within this document.

AES – Advanced Encryption Standard
ASCII – American Standard Code for Information Interchange
ASN.1 – Abstract Syntax Notation One (ASN.1)
ATSC – Advanced Television Systems Committee
BMFF – Base Media File Format
CA – Certificate Authority
CD – Companion Device
CENC – Common ENCryption
CMS – Cryptographic Message Syntax
CRL – Certificate Revocation List
CTA – Consumer Technology Association
DASH – Dynamic Adaptive Streaming over HTTP
DASH-IF – DASH Industry Forum
DNS – Domain Name System
DNSSEC – Domain Name System Security Extensions
DRM – Digital Rights Management
ECDHE – Elliptic Curve Diffie-Hellman Ephemeral key exchange
ECDSA – Elliptic Curve Digital Signature Algorithm
GCM – Galois Counter Method
IANA – Internet Assigned Numbers Authority
IETF – Internet Engineering Task Force
IKM – Input Keying Material
IP – Internet Protocol
ISO – International Organization for Standardization
LLS – Low Level Signaling
MIME – Multipurpose Internet Mail Extensions
MMT – MPEG Media Transport
MPEG – Moving Pictures Experts Group
OCSP – Online Certificate Status Protocol
PD – Primary Device
PIN – Personal Identification Number
PKI – Public Key Infrastructure
RFC – Request for Comments

ROUTE – Real-Time Object Delivery over Unidirectional Transport

RSA – A method for obtaining digital signatures and public-key cryptosystems (originally proposed by Rivest, Shamir, and Adelman).

SECP – Standard for Efficient Cryptography Elliptic Curve Domain Parameters

SHA – Secure Hash Algorithm

S/MIME – Secure/Multipurpose Internet Mail Extensions

TLS – Transport Layer Security

UUID – Universally Unique Identifier

W3C – Worldwide Web Consortium

XML – eXtensible Markup Language

3.4 Terms

The following terms are used within this document.

ATSC 3.0 Server – Any IP-connected device that provides content or other service to an ATSC 3.0 client, and that complies with the normative requirements of this standard.

Author Signature – A signature encoded in the form specified in Section 5.2 below that is generated by the author of the application, which is the entity or entities that claim authorship over the application content.

Certificate Authority – An entity that issues digital certificates.

Cipher Suite – A suite of cryptographic algorithms used together.

CMS Signed Data – See Section 5.2.2.2.

Companion Device – See A/338 [29].

Cryptographic Message Syntax – The message defined by RFC 5652 [13].

Distributor Signature – A signature encoded in the form specified in Section 5.2 below that is generated by a distributor, which is a third party (e.g., the broadcaster) that is distributing the application on behalf of the author.

Elliptic Curve Group – See TLS 1.3 [23].

Extended Key Usage extension – See RFC 5280 [10].

Hash Algorithm – a one-way mathematical algorithm, which is infeasible to invert, that maps data of arbitrary size to a hash of a fixed size.

Key Usage extension – See RFC 5280 [10].

LLS Table – Low-Level Signaling Table, see A/331 [26].

Message Digest Algorithm – Hash Algorithm

OCSP Responder – A server typically run by the certificate issuer that returns an OCSP Response

OCSP Responder Identifiers – A list of SHA-1 hashes, one hash for each trusted OCSP Responder public key

OCSP Response – The response to an OCSP request, see RFC 6960 [21].

Pre-Shared Key Exchange Mode – See RFC 8446 [23], Section 4.2.9

Pre-Shared Key Exchange Parameters – The parameters defined in Section 5.6.2.

Primary Device – The source device to a companion device

Private Enterprise Number – An ITU-T X.660 Object Identifier allocated to a private organization, such as ATSC.

Privileged Application – An application that can override system controls, authorizations, or privileges.

Public Key Infrastructure – A public key infrastructure (PKI) is a set of roles, policies, and procedures needed to create, manage, distribute, use, store & revoke digital certificates and manage public-key encryption.

Secure Connection – An IP connection secured by TLS, as described in Section 5.1.

Server Name Indication extension - See RFC 6066 [19].

Service Level Signaling – Signaling which provides information for discovery and acquisition of ATSC 3.0 Services and their content components.

Signature Algorithm - a mathematical scheme for verifying the authenticity of digital objects

X.509 Certificates – digital certificates that use the ITU-T X.509 public key infrastructure (PKI) standard to verify that a public key belongs to the user, computer or service identity contained within the certificate.

reserved – Set aside for future use by a Standard.

3.5 Extensibility

The protocols specified in the present Standard are designed with features and mechanisms to support extensibility. In general, the mechanisms include:

- Use of "protocol version" fields
- Definition of fields and values reserved for future use
- Use of XML, which is inherently extensible by means of future addition of new attributes and elements, potentially associated with different namespaces

Receiving devices are expected to disregard reserved values, and unrecognized or unsupported descriptors, XML attributes and elements.

3.6 XML Schema and Namespace

A number of new XML elements are defined and used in this Standard. These elements provide various Service signaling elements and attributes defined in this Standard (see for example Section 5.2.2.2). These new XML elements are defined with separate namespaces in schema documents that accompany this Standard. The namespaces used by various schemas are described in individual sections of the present document. The sub-string part of namespaces between the right-most two '/' delimiters indicate major and minor version of the schemas. The schemas defined in this present document shall have version '1.0', which indicates major version is 1 and minor version is 0.

The namespace designator, "`xs:`", and many terms in the "Data Type" column of tables is a shorthand for datatypes defined in W3C XML Schema [26]] and shall be as defined there.

In order to provide flexibility for future changes in the schema, decoders of XML documents with the namespaces defined in the present document should ignore any elements or attributes they do not recognize, instead of treating them as errors.

All element groups and attribute groups are explicitly extensible with elements and attributes, respectively. Elements can only be extended from namespaces other than the target namespace. Attributes can be extended from both the target namespace and other namespaces. If the XML schema does not permit this for some element, that is an error in the schema.

XML schemas shall use `processContents="strict"` in order to reduce inadvertent typos in instance documents. Further, users are encouraged to modify all referenced third-party schemas to change `processContents` to `"strict"`.

XML instance documents shall use UTF-8 encoding.

In the event of any discrepancy between the XML schema definitions implied by the tables that appear in this document and those that appear in the XML schema definition files, those in the XML schema definition files are authoritative and take precedence.

The XML schema document for the schemas defined in this document can be found at the ATSC website.

4. SYSTEM OVERVIEW

4.1 Features

This specification defines a set of methods designed to secure the following content and data flows described in other ATSC 3.0 specifications:

1) Content protection for MPEG-DASH content delivery (Section 5.7)
2) Authentication of ATSC 3.0 applications (Section 5.2)
3) Authentication of ATSC 3.0 Broadcast Signaling (Section 5.3)
4) Interactive data exchanged over an internet connection between an ATSC 3.0 application and a web content server (Section 5.1), including the use of DNS Security (Section 5.1.1.7)
5) Data flows between an ATSC 3.0 primary device and a companion device (Section 5.6)

4.2 System Architecture

This specification defines a number of profiles for established security specifications defined by IETF, ISO and W3C. In defining these profiles, this specification seeks to establish a consistent use of cryptographic algorithms across the different content and data flows that it addresses. The profiles are designed to provide some degree of flexibility in the choice of cryptographic algorithms being used in a particular flow while enabling the use of commonly available implementations of the specified standard technologies.

In the case of MPEG-DASH content protection, this specification defines the use of common encryption techniques that allow content protection licences to be delivered to a number of different content decryption modules from different suppliers.

4.3 Central Concepts

Several of the specifications referenced herein make use of a chain of trust based on the provisioning of X.509 certificates in the message flow and the establishment of a set of trust anchors within the ATSC 3.0 receiver (Sections 5.2.2 and 5.4). In addition to the concept of the chain of trust, this specification also defines the carriage of certificate revocation information in On-line Certificate Status Protocol (OCSP) response constructs in order to verify the validity of the certificates in the chain of trust (Section 5.5). The carriage of these constructs within the message flow avoids each ATSC 3.0 receiver separately requesting this information thus avoiding unnecessary traffic flow peaks to the OCSP responder.

5. SPECIFICATION

5.1 Transport Protection

Transport protection provides protection against spoofing or hijacking the delivery of the data. This may include protection of content that is not separately encrypted. Encryption of content in transit will be described in this section.

5.1.1 Internet Streaming Transport Security

5.1.1.1 TLS – Transport Layer Security

ATSC 3.0 clients are expected to implement both TLS 1.3 [23] and TLS 1.2 (RFC 5246 [9]) for secure connections over broadband. An ATSC 3.0 client is expected to request a connection using TLS 1.3 (ProtocolVersion { 0x03, 0x04 }), but is also expected to accept a server's request to downgrade the connection to TLS 1.2 (ProtocolVersion { 0x03, 0x03 }) in the manner specified in TLS 1.3 [23] Appendix D.

An ATSC 3.0 Server, when negotiating a secure connection for use with ATSC 3.0 broadband protocols, should comply with TLS 1.3. An ATSC 3.0 Server that does not support TLS 1.3 shall respond with a TLS 1.3 [23] Server Hello message specifying a ProtocolVersion { 0x03, 0x03 } (indicating TLS 1.2). The server shall refuse Secure Connection negotiations with clients that do not support a ProtocolVersion equal to or greater than { 0x03, 0x03 } and shall send a protocol_version alert message to the client as described in TLS 1.3 [23] Appendix D (TLS 1.2 [9] Appendix E).

5.1.1.2 TLS 1.3 Server Connection Negotiation

An ATSC 3.0 Server that supports TLS 1.3 shall only negotiate Secure Connections using one or more combinations of a Cipher Suite, Elliptic Curve Group, and Signature Algorithm as specified in Sections 5.1.1.2.1, 5.1.1.2.2 and 5.1.1.2.3 respectively.

ATSC 3.0 Servers that support TLS 1.3 shall decline to establish a connection that does not request at least one combination of these Signature Algorithms, Elliptic Curve Groups, and Cipher Suites.

ATSC 3.0 clients that support TLS 1.3 are expected to only negotiate Signature Algorithms, Elliptic Curve Groups, and Cipher Suites identified in this section.

5.1.1.2.1 Cipher Suites

```
TLS_AES_128_GCM_SHA256
TLS_AES_256_GCM_SHA384
TLS_CHACHA20_POLY1305_GCM_SHA256
```

(as specified in TLS 1.3 [23]).

5.1.1.2.2 Elliptic Curve Groups

```
secp256r1
secp384r1
secp521r1
```

(as specified in TLS 1.3 [23]).

Each Elliptic Curve Group shall be used with the uncompressed point format.

5.1.1.2.3 Signature Algorithms

```
rsa_pkcs1_sha256
rsa_pkcs1_sha384
rsa_pkcs1_sha512
ecdsa_secp256r1_sha256
ecdsa_secp384r1_sha384
ecdsa_secp521r1_sha512
rsa_pss_rsae_sha256
rsa_pss_rsae_sha384
rsa_pss_rsae_sha512
```

(as specified in TLS 1.3 [23]).

5.1.1.3 TLS 1.2 Server Connection Negotiation

ATSC 3.0 Servers that only support TLS 1.2 shall negotiate Secure Connections using one or more of the following Cipher Suites (as specified in RFC 5289 [11]):

```
TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384
TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
```

or one or more of the following Cipher Suites (as specified in RFC 7539 [24]) where these Cipher Suites are requested by the client:

```
TLS_ECDHE_ECDSA_WITH_CHACHA20_POLY1305_SHA256
TLS_RSA_ECDSA_WITH_CHACHA20_POLY1305_SHA256
```

or

```
TLS_RSA_WITH_AES_128_CBC_SHA
```

(as specified in RFC 5246 [9]) may be negotiated for; however, the server shall only choose this Cipher Suite as the least preferred of the client's Cipher Suites (irrespective of the order supplied by the client).

5.1.1.3.1 Elliptic Curve Groups

An ATSC 3.0 Server shall support the following Elliptic Curve Groups: secp256r1, secp384r1, and secp521r1. An ATSC 3.0 Server shall support the uncompressed point format.

Servers shall decline to establish a connection that does not request one or more of these Elliptic Curve Groups or point formats.

The client is expected to only negotiate Elliptic Curve Groups and point formats that are required to be supported by an ATSC 3.0 Server.

5.1.1.3.2 Signature Algorithms

An ATSC 3.0 Server shall support the rsa or ecdsa Signature Algorithm with any of sha256, sha384 or sha512 Hash Algorithm.

An ATSC 3.0 client that is negotiating (or renegotiating) a TLS 1.2 connection may request one of these Signature Algorithm and Hash Algorithm combinations or may omit the TLS 1.3 [23]

Signature Algorithm extension. When a client does not include a Signature Algorithm extension, the ATSC 3.0 Server shall reject the connection request with an insufficient_security error.

5.1.1.4 Server Certificate Selection

An ATSC 3.0 Server shall only supply certificates with signatures using one of the supported signature and hash algorithm combinations (see Sections 5.1.1.3.2 above) that is negotiated by the client (even in the case that the client attempts to negotiate other algorithms) and shall not establish a Secure Connection with certificates that use other algorithms.

When a client requests a connection over TLS 1.2 or TLS 1.3 the client is expected to include a Server Name Indication extension as specified in RFC 6066 [19] that contains the fully qualified DNS host name of the server. The ATSC 3.0 Server shall use the Server Name Indication provided by the client to assist in the selection of a suitable server certificate to return to the client in the TLS handshake.

When a client requests a connection over TLS 1.3 the client can include a Certificate Authorities extension as specified in TLS 1.3 [23] to provide a list of the trusted root certificates that it holds in its secure store. When a client requests a connection over TLS 1.2 the client can include a Trusted CA Indication extension as specified in RFC 6066 [19] to provide a list of the trusted root certificates that it holds in its secure store. Receiver manufacturers choose the set of trusted root certificates. The ATSC 3.0 Server shall use the Trusted CA Indication extension to assist in the selection of a suitable certificate chain to return to the client in the TLS handshake.

In the case that an ATSC 3.0 Server is unable to select a certificate chain that matches the client criteria in either the Server Name Indication extension or the Trusted CA Indication extension, the ATSC 3.0 Server shall not establish the connection.

5.1.1.5 TLS Certificate Status Request and Response

The client is expected to include the Certificate Status Request extension as specified in RFC 6066 [19] Section 8. The Certificate Status Request extension includes a list of OCSP Responder Identifiers each encoded as a SHA-1 hash of the trusted OCSP responder public key as defined in RFC 6960 [20]. An ATSC 3.0 Server shall only supply to the client the OCSP Responses that the ATSC 3.0 Server has received from OCSP responders with responder public keys that are trusted by the client and which are signed using signature algorithms supported by the client. If an ATSC 3.0 Server is unable to obtain an OCSP Response for a certificate that the server supplies from an OCSP Responder that is identified by the client as a trusted responder, the ATSC 3.0 Server shall not establish the connection.

The ATSC 3.0 Server shall forward the most recent OCSP Response (see Section 5.5.1 below) for the certificates the server uses to establish a connection to the ATSC 3.0 client. The format of the OCSP Response provided by the responder should be limited to the mandatory elements defined in RFC 5019 [7] and no optional elements should be included in the response. When a server is establishing a connection over TLS 1.2, the server shall include the OCSP Response in its Certificate Status handshake message (immediately after its Certificate handshake message) as defined in RFC 6066 [19]. When a server is establishing a connection over TLS 1.3, the server shall include the OCSP Response in the Certificate message.

The ATSC 3.0 client is expected to verify the Certificate Status message provided by the server as specified in RFC 6066 [19] Section 8. A client uses the OCSP Response data that it receives to verify that the certificates that authenticate server connections are valid at the time the connection is established. See CTA 2053 [28].

5.1.1.6 TLS Session Resumption

An ATSC 3.0 Server that has a newly established TLS 1.3 connection may provide a TLS 1.3 [23] New Session Ticket message once the server has received the client's TLS 1.3 [23] Finished handshake message. The New Session Ticket message shall not include the TLS 1.3 [23] Early Data Indication extension. A client may supply the information from this session ticket in the TLS 1.3 [23] **`pre_shared_key`** extension in a subsequent TLS 1.3 [23] Client Hello message to resume the TLS session. The client is expected to negotiate session resumption using the same Elliptic Curve Group and Cipher Suite and Server Name Indication extension as used when the original connection was established. The client is expected to set the Pre-Shared Key Exchange Mode set to `psk_dhe_ke` which will enable a new ephemeral ECDHE key to be established.

On receipt of a session resumption TLS 1.3 [23] Client Hello the ATSC 3.0 Server shall verify that the session ticket is still valid and that the client has selected the same Elliptic Curve Group and Cipher Suite as used for the original connection. The server shall also verify that the Server Name Indication extension supplied in the TLS 1.3 [23] Client Hello message is the same as that provided for the original connection. The server shall only negotiate a session resumption request that includes a Pre-Shared Key Exchange Mode set to `psk_dhe_ke`.

The ATSC 3.0 Server shall not respond to a TLS 1.3 [23] Client Hello message that contains early_data thus requiring the client to issue a session resumption TLS 1.3 [23] Client Hello message without any early data.

An ATSC 3.0 Server that has established a TLS 1.2 connection session may support the Session Ticket extension (RFC 5077 [8]) to allow later resumption of that session. If the ATSC 3.0 Server does not support this extension, then the server shall not send an empty Session Ticket extension to the client that has requested session ticket information.

5.1.1.6.1 TLS Connection Renegotiation

TLS 1.3 does not support connection renegotiation.

An ATSC 3.0 client that is processing a TLS 1.2 handshake is expected to support the Renegotiation Indication extension (RFC 5746 [14]) but is not expected to send a TLS 1.3 [23] Client Hello handshake message that includes any data in this extension. An ATSC 3.0 Server that is processing a TLS 1.2 handshake shall include an empty Renegotiation Indication extension as required by RFC 5746 [14] in the TLS 1.3 [23] Server Hello message to indicate that it does not support renegotiation. An ATSC 3.0 Server that is processing a TLS 1.2 handshake shall not send a TLS 1.3 [23] Server Hello message to the client to instigate renegotiation of connection parameters.

5.1.1.7 DNSSEC – Domain Name System Security Extensions

An ATSC 3.0 Server shall be a member of a DNSSEC signed zone as described in RFC 6840 [20] and RFC 4033 [5]. This specification expects that an ATSC 3.0 receiver implements a DNSSEC Security-Aware Stub Resolver as specified in RFC 4033 [5].

5.2 ATSC 3.0 Cryptographic Signing

This standard includes mechanisms below for cryptographically signing applications and signaling. Implementation of these features requires one or several Public Key Infrastructure(s) (PKI) that provide certificates aligned to the profiles specified in Section 5.3 and that are supported by the inclusion of associated root certificate(s) in receivers, all of which is out of scope of this document.

5.2.1 ATSC 3.0 Application Code Signing

Executable or interpretable code shall be packaged as a multi-part MIME package and shall be cryptographically signed.

Signed applications shall be formatted as specified in S/MIME Version 3.2 (RFC 5751 [15]) as follows:

1) An Author Signature shall be added first in the manner specified in S/MIME [15] Section 3.4.3 to create a detached signature. The name attribute for the newly created Content Type application/pkcs7-signature shall be set to author.p7s and the filename attribute for the corresponding Content Disposition shall be set to author.p7s. The Author Signature shall only appear as the first detached signature in the final MIME package.
2) A Distributor Signature shall then be added in the manner specified in S/MIME [15] Section 3.4.3 to create a detached signature. The output MIME package from that Author Signature process becomes the input to this step of the process. The name attribute for the newly created Content Type application/pkcs7-signature shall be set to distrib.p7s and the filename attribute for the corresponding Content Disposition shall be set to distrib.p7s. The Author Signature shall appear as the first detached signature in the final MIME package, and the Distributor Signature shall appear as the second detached signature in the final MIME package.
3) Any compression shall be applied after each of the signatures has been included in the multi-part MIME package. The signatures generated using S/MIME processing shall be encoded according to the Cryptographic Message Syntax (RFC 5652 [13]) with the extension for elliptic curve signature processing as defined in RFC 5753 [16]. Each CMS block shall include an End-Entity certificate that authenticates the signature and a set of any Intermediate Certificate Authority certificates that authenticate issuer(s) of the certificates included in the CMS block.

The following profile shall be used to create the S/MIME digital signature:

1) The Signature Algorithm and Message Digest Algorithm shall be one of the following pairs:
   - rsa- pkcs1 with sha-256
   - ecdsa curve secp256r1 with sha-256
   - ecdsa curve secp384r1 with sha-384
   - ecdsa curve secp521r1 with sha-512
2) The RFC 5652 [13] SignerInfo Type shall contain a SigningTime attribute that shall contain the time at which the signature is generated as specified in S/MIME [15] Section 2.5. This attribute shall be encoded as a signed attribute.

5.2.2 ATSC 3.0 Signaling Message Signing

5.2.2.1 Overview

ATSC 3.0 service signaling is carried in a number of different types of message each of which can include a CMS Signed Data structure (RFC 5652 [13] with the extension for elliptic curve signature processing as defined in RFC 5753 [16]) that provides a verifiable signature for the message content. The basic characteristics of each CMS Signed Data structure are as follows:

1) The RFC 5652 [13] SigningTime attribute containing the time at which the signature is generated is included in the RFC 5652 [13] SignerInfo structure as a signed attribute.
2) The SubjectKeyIdentifier is included as the SignerIdentifier.

3) No Encapsulated Content, no Certificates and no CRLs are included.
4) The Signature Algorithm and Message Digest Algorithm shall be one of the following pairs:
   - rsa- pkcs1 with sha-256
   - ecdsa curve secp256r1 with sha-256
   - ecdsa curve secp384r1 with sha-384
   - ecdsa curve secp521r1 with sha-512

(Additional characteristics are defined in each usage definition in subsequent sections.)

In addition a CertificationData table is defined below to be carried in the low-level signaling. The CertificationData table carries the necessary information for the authentication to a known root certificate and status verification of the keys used to sign signaling message content. The CertificationData table also carries information that allows the broadcaster to:

1) Manage a change of the signaling message signing key,
2) Define the life-span of certificate status response information, and
3) Request the receiver to handle signature verification failures in a particular manner.

5.2.2.2 Certificate and OCSP Response LLS Table

This specification defines a new LLS Table that carries X.509 Certificates and OCSP Responses that are used to verify signed signaling tables.

When one or more signaling tables are signed, the CertificationData LLS Table shall be included among the LLS Tables described in ATSC A/331 [26] Section 6.1, and shall use LLS_table_id 0x06.

The CDT shall be represented as an XML document containing a CertificationData root element that conforms to the definitions in the XML schema that has namespace:

tag:atsc.org,2016:XMLSchemas/ATSC3/Delivery/CDT/1.0/

The definition of this schema is in an XML schema file, *CDT-1.0-20200229.xsd*, accompanying this Standard, as described in Section 3.6 above.

Note that the CertificationData LLS Table is a standalone table that contains its own signature (i.e., is not in a signed_multitable message), as the data in the CertificationData LLS Table is required to verify the signature of a signed_multitable message. Note also that the attributes, certificates, and OCSP Responses carried in the CertificationData LLS Table are unrelated to application signing (Section 5.2.1), which has different requirements and a different mechanism for carrying certificates, OCSP Responses and related data.

The XML schema xmlns short name should be "cdt". The CertificationData LLS Table has the following informative description:

Table 5.1 CertificationData XML Format

Element or Attribute Name	Use	Data Type	Short Description
`CertificationData`			Root element of the CertificationData table.
`ToBeSignedData`	1		
`@OCSPRefresh`	1	xs:dayTimeDuration	The duration for which an OCSPResponse carried in this Certification Data is considered valid from its producedAt time.
`Certificates`	1..N	Base64 String	A list of certificates that are used to authenticate a broadcaster signature. This must include end-entity certificates authenticating the CurrentCert and the CMSSignedData signing certificate and any intermediate CA certificates used to validate these certificates. The Root CA certificate is not included in the list.
`CurrentCert`	1	Base64 String	SubjectKeyIdentifier for the certificate currently used to sign signaling messages.
`CertReplacement`	0..1		
`@NextCertFrom`	1	xs:dateTime	Earliest time at which NextCert can be validly used.
`@CurrentCertUntil`	1	xs:dateTime	Latest time at which CurrentCert can be validly used.
`NextCert`	1	Base64 String	SubjectKeyIdentifier for the certificate next used to sign signaling messages.
`CMSSignedData`	1	Base64 String	A CMS Signed Data structure authenticating the ToBeSignedData contained in this table.
`OCSPResponse`	1..N	Base64 String	A set of OCSP Responses that provide status information for each of the Certificates carried in this Certification Data.

CertificationData – Root element of the CertificationData LLS Table.

ToBeSignedData – The data elements to be included in the signature calculation contained in the CMSSignedData element. The signature contained in CMSSignedData is across all data, including the beginning and ending tags of this field (from the initial "<" through the final ">").

Certificates – A list of X.509 certificates matching the profile specified in Section 5.3.1.6 (or Section 5.3.1.3 in the case of CA certificates) each of which is encoded as a base64 string. The list shall include the following certificates:

1) An end-entity certificate that is referenced by CurrentCert.
2) An end-entity certificate that is referenced from CMSSignedData with the same SubjectName as the CurrentCert. The broadcaster should protect the key authenticated by this certificate independently from the key authenticated by CurrentCert, preferably in an environment that prohibits internet access.
3) If a CertReplacement element is included, the end-entity certificate that is referenced by NextCert with the same SubjectName as the CurrentCert.
4) The set of Certificate Authority certificates that authenticate the issuers of other certificates in this list.

CurrentCert – The SubjectKeyIdentifier for the certificate that is currently used to sign signaling messages.

CertReplacement – An optional element that is used to indicate the replacement of CurrentCert and the timeframe during which that replacement will take place.

NextCert – The SubjectKeyIdentifier for the certificate that will replace the CurrentCert and be used to sign signaling messages.

@NextCertFrom – The date and time from which the broadcaster can validly sign signaling messages using the NextCert.

@CurrentCertUntil – The date and time until which the broadcaster can validly sign signaling messages using the CurrentCert. Note that this may be later than the NextCertFrom date, but cannot be earlier than that date.

@OCSPRefresh – The duration after which an OCSP Response is considered to be invalid, based on the producedAt time in the response structure and the current system time. This field shall not exceed a duration of ten days (two hundred forty hours), except for test/validation (e.g., with a test/validation indicator) or similar usage, and should not include fractional seconds. Practically, **@OCSPRefresh** should be at least one hour. But note that this value is related to vulnerability periods, see for example, Sec. 4.9.10 of [30], which limits the expiration time of certain OCSP Responses to ten days.

CMSSignedData – The CMS Signed Data (RFC 5652 [13]) element with the following characteristics:

1) The characteristics specified in Section 5.2.2.1 above.
2) The content being signed shall be the full extent of the ToBeSignedData element.
3) The SubjectKeyIdentifier shall identify an end-entity certificate in Certificates other than that identified by CurrentCert, and other than that identified by NextCert if present.

OCSPResponse – A set of one or more OCSP Response structures in the form specified in RFC 6960 [21] that provide certificate status information for the Certificates carried in this CertificationData. Each **OCSPResponse** in the set may contain a number of OCSP Single Response (see RFC 6960 [21]) structures where the same OCSP Responder is authorised to issue a response for more than one of the Certificates.

5.2.2.3 Signatures for Low Level Signaling (LLS) Tables

A signature that is applied to a LLS message is carried in a CMS Signed Data (RFC 5652 [13]) element with the following characteristics:

1) The characteristics shall be as specified in Section 5.2.2.1 above.
2) The SignerIdentifier shall match either the CurrentCert or, if present, the NextCert.

5.2.2.4 Signatures for Service Level Signaling carried over ROUTE/DASH

Service Level Signaling over ROUTE/DASH is encapsulated in multi-part MIME packages and the broadcaster signs each of these packages in the manner specified in S/MIME [15] Section 3.4.3 with the CMS Signed Data structure profile as specified below to create a detached signature. The name attribute for the newly created Content Type application/pkcs7-signature shall be set to bcsig.p7s and the filename attribute for the corresponding Content Disposition shall be set to bcsig.p7s.

The signatures generated using S/MIME processing shall be encoded according to the Cryptographic Message Syntax (RFC 5652 [13]). The following profile for the CMS Signed Data structure shall be used to create the S/MIME digital signature:

1) The characteristics specified in Section 5.2.2.1 above.
2) The SignerIdentifier shall match either the CurrentCert or, if present, the NextCert.

All Service Level Signaling encapsulated in multi-part MIME packages shall be signed by the broadcaster.

5.2.2.5 Signatures for MMT Messages

The broadcaster signature of an MMT message is across the entire MMT message (not including the signature), and shall be carried in a CMS Signed Data (RFC 5652 [13]) structure with the following characteristics:

1) The characteristics specified in Section 5.2.2.1 above.
2) The SignerIdentifier shall match either the CurrentCert or, if present, the NextCert.

5.2.2.6 Receiver Signature Verification of Signaling Messages (Informative)

To help identify suspected stream tampering, the receiver is expected to undertake the following tasks to verify new instances of the signed signaling messages described in Sections 5.2.2.3, 5.2.2.4, and 5.2.2.5:

1) Verify that the signature contained in the CMS Signed Data structure is correct.
2) Verify that the SigningTime attribute in the CMS Signed Data structure is:
   a) Not greater than the System Time, and
   b) Not less than the SigningTime of any previously received instance of the same type of signed signaling message. For example, the SigningTime sequence would not go backward in time for any signaling within a single message type of (i) SLS of the same specific service, or (ii) an LLS CDT message on the same RF channel or bonded channels, or (iii) an LLS SignedMultiTable message on the same RF channel with the same set of LLS tables within.
3) Verify that the key used to sign the signaling message is authenticated by an unexpired end-entity certificate carried in CertificationData message,
   a) verify that this certificate has an extended key usage that includes id-atsc-kp-signalingSigning, and
   b) verify that this certificate contains a Subject Directory Attribute extension with an attribute of type id-atsc-sdattr-bsid and values that contain a SET OF INTEGER (as described in RFC 5280 [10]), and that the set of values matches exactly the set of bsids listed in the Service List Table for this broadcast stream, (e.g., when identifying a bonded pair of RF channels, both the bsids listed in the SLT and the set of bsids in the certificate identify exactly the same two bsid values), and
   c) verify that this end-entity certificate's SubjectKeyIdentifier matches either the CurrentCert or, if present, the NextCert.
4) Verify that, at the SigningTime of the signaling message, the CurrentCert or NextCert used to authenticate the signing key was valid for use according to the CurrentCertUntil and NextCertFrom dates, respectively.
5) Verify that the producedAt date in the OCSPResponse that provides status information for the end-entity certificate plus the number of hours specified as the OCSPRefresh period in the CertificationData message exceeds the current System Time.

The receiver is expected to undertake the following tasks to verify a new instance of the CertificationData LLS Table described in Section 5.2.2.2:

1) Verify each of the certificate chains carried in the CertificationData message and that the first Certificate Authority certificate in that chain is issued by a Root Certificate Authority trusted by the receiver.
2) Verify that each of the certificates in the authenticating certificate chain has a status of good in the OCSP Response.

3) Verify that the signature contained in CMSSignedData in the CertificationData LLS message is valid and authenticated by a certificate chain in that message, and that the certificates in this chain have a status of good in the OCSPResponse corresponding to that certificate. Also verify that the key pair used to sign the CertificationData message is different from either of the key pairs that are designated for use in signing other signaling messages as indicated by the CurrentCert and, if present, NextCert elements.
4) Verify that OCSPRefresh is not greater than ten days (two hundred forty hours) and that the producedAt date in each OCSPResponse plus the number of hours specified as the OCSPRefresh period in the CertificationData message exceeds the current System Time.
5) Revalidate the next instance of each signed signaling message that is received after the CertificationData LLS Table is successfully verified.

5.2.2.7 CMS Signed Data XML structure

Where CMS Signed Data is transmitted as an XML structure, the characteristics shall be as specified in Section 5.2.2.1 and shall be represented as an XML document containing a CMSSignedData root element that conforms to the definitions in the XML schema that has namespace:

tag:atsc.org,2016:XMLSchemas/ATSC3/Delivery/CMSSD/1.0/

The definition of this schema is in an XML schema file, *CMSSD-1.0-20200229.xsd*, accompanying this Standard, as described in Section 3.6 above.

The XML schema xmlns short name should be "cmssd". The informative definition of this XML schema is as follows:

Table 5.2 CMS Signed Data XML Format

Element or Attribute Name	Use	Data Type	Short Description
CMSSignedData	1	Base64 string	A base64 encoded encapsulation of the CMS Signed Data structure (RFC 5652 [13])

Any data compression shall be applied after the CMS Signed Data XML document has been appended to the message.

5.3 Certificates and Certificate Management

This standard uses the Internet X.509 Public Key Infrastructure Profile (RFC 5280 [10]) as the base profile for certificates used by an ATSC 3.0 TLS server and ATSC 3.0 application and signaling signing authority authentication.

The following types of certificates are used by ATSC 3.0 devices during the authentication process:

- One or more root certificates. These are trusted self-signed certificates issued by a trusted certificate authority as the root of trust. Each certificate path validation process completes when a trusted root certificate is reached. TLS does not require the signature contained within these certificates to be checked.
- Certificate authority certificates. These certificates are issued by a trusted root certificate authority or a certificate authority whose certificate path can be validated to a trusted root certificate authority.

- TLS server certificates. These certificates are issued by a trusted certificate authority and are designated for use in server authentication.
- ATSC 3.0 author and distributor application signer certificates. These certificates are issued by a trusted certificate authority and are designated for use in code signing.
- ATSC 3.0 broadcast signaling signer certificates. These certificates are issued by a trusted certificate authority and are designated for use in signing broadcast signaling messages.
- OCSP responder certificates. These certificates are issued by a trusted certificate authority and are designated for use in OCSP responder authentication.

The client is expected to perform certificate chain validation as specified in RFC 5280 [10] using the certificate status information provided by the ATSC 3.0 Server in stapled OCSP Response messages (see Sections 5.1.1.5 and 5.5.2) as a reliable source for revocation information.

5.3.1 Certificate Profiles

The profile specified in RFC 5280 [10] is further constrained for certificates used in ATSC 3.0.

5.3.1.1 General

All ATSC 3.0 certificates shall be X.509 version 3 certificates.

All keys contained in ATSC 3.0 certificates shall be either RSA keys with a minimum size of 2048 bits encoded as specified in RFC 3279 [4] or ECDSA keys which use the elliptic curve groups and point format defined above (Sections 5.1.1.2 and 5.1.1.3) and encoded as specified in RFC 5480 [12].

All RSA signatures contained in ATSC 3.0 certificates shall be encoded according to the RSA signature algorithms specified in RFC 3279 [4] and RFC 4055 [5].

All ECDSA signatures contained in ATSC 3.0 certificates shall be encoded according to the ECDSA signature algorithms specified in RFC 5758 [17] and shall use one of the hash algorithms specified above (Sections 5.1.1.1 and 5.1.1.3) for use with the ECDSA signature algorithm.

All ATSC 3.0 end-entity certificates shall contain a Key Usage extension containing at least the digitalSignature value. All ATSC 3.0 certificates shall use algorithms and identifiers with values constrained as specified in RFC 3279 [4] and RFC 4055 [5].

ATSC 3.0 devices need not process the RFC 5280 [10] Authority Information Access extension or the Subject Information Access extensions.

5.3.1.2 Root Certificate Profile

The RSA key size for any root certificate shall be at least 2048 bits and should be 4096 bits.

The ECDSA key size for any root certificate shall be at least 384 bits.

5.3.1.3 Certificate Authority Certificate Profile

The RSA key size for any certificate authority certificate shall be at least 2048 bits.

The ECDSA key size for any certificate authority certificate shall be at least 256 bits.

5.3.1.4 Server Authentication Certificate Profile

The RSA key size for this certificate shall be at least 2048 bits.

The ECDSA key size for any server authentication certificate shall be at least 256 bits.

The RFC 5280 [10] Subject Alternative Name extension shall be present and shall include either the DNS Name or the IP Address of the server being authenticated.

The Extended Key Usage extension shall be present and shall be set to the value id-kp-serverAuth to indicate that the certificate is used in TLS server authentication.

5.3.1.5 ATSC 3.0 Application Signer Certificate Profile

The RSA key size for any application signer certificate shall be at least 2048 bits.

The ECDSA key size for any application signer certificate shall be at least 256 bits.

The Key Usage extension shall be marked as critical and shall include only the digitalSignature value.

The Extended Key Usage extension shall be present, marked as critical, and shall include the value id-kp-codeSigning to indicate that the certificate is used in the signing of downloadable executable code. For author code signing certificates this extension shall also include the value id-atsc-kp-author. For distributor code signing certificates this extension shall include the value id-atsc-kp-distributor.

For distributor code signing certificates, Subject Directory Attributes extension shall be present, not marked as critical, and shall include an attribute of type id-atsc-sdattr-bsid and values that contain a SET OF INTEGER (as described in RFC 5280 [10]), each integer in the set contains a Broadcast Stream Identifier.

5.3.1.6 ATSC 3.0 Broadcast Signaling Signer Certificate Profile

The RSA key size for any broadcast signaling signing certificate shall be at least 2048 bits.

The ECDSA key size for any broadcast signaling signing certificate shall be at least 256 bits.

The Key Usage extension shall be marked as critical and shall include only the digitalSignature value.

The Extended Key Usage extension shall be present, shall be marked as critical, and shall include an attribute of type id-atsc-kp-signalingSigning to indicate that the certificate is used in the signing of ATSC signaling constructs.

The Subject Directory Attributes extension shall be present, not marked as critical, and shall include an attribute of type id-atsc-sdattr-bsid and values that contain a SET OF INTEGER (as described in RFC 5280 [10]), each integer in the set contains a Broadcast Stream Identifier.

5.3.1.7 OCSP Responder Certificate Profile

The RSA key size for any OCSP responder certificate shall be at least 2048 bits.

The ECDSA key size for any OCSP responder certificate shall be at least 256 bits.

The Extended Key Usage extension shall be present and shall be set to the value id-kp-OCSPSigning to indicate that the certificate is used to sign OCSP Responses.

5.4 ATSC 3.0 Client Certificate Storage

See CTA 2053 [28], which describes secure storage of certificates, and the mechanism(s) for modifying certificates used by client devices.

Clients provide secure storage for the following set of certificates:

- The set of trusted root certificates
- The set of trusted signing certificate authority certificates
- The set of trusted OCSP responder certificates

Certificates are changed over time, either by client device code download or by other means.

5.5 Certificate Revocation and Status Information

The management of certificate status is under the control of the issuing authority which works according to their defined certification practices and policies. Each certificate authority that issues certificates used by an ATSC 3.0 Server or ATSC 3.0 application signing authority is responsible

for the timely supply of certificate status information to the OCSP responder(s). The specific methods by which this information is made available to the OCSP responder are beyond the scope of this specification.

5.5.1 Certificate Revocation and Status Information for TLS Server Certificates

An ATSC 3.0 Server shall request certificate status information from an OCSP responder at least once per minute for each server authentication certificate that it provides as server identification when establishing a TLS connection. The request shall be in the format specified in RFC 6960 [20], shall be unsigned, and the only RFC 6960 extension included in the request shall be the Preferred Signature Algorithms extension.

> Note: In order to satisfy clients that support different signature algorithms, a server may need to request certificate status information from the same OCSP responder using different values in the RFC 6960 Preferred Signature Algorithms extension.

5.5.2 Certificate Revocation and Status Information for ATSC 3.0 Application Signing Certificates

An ATSC 3.0 application signing authority shall request certificate status information from an OCSP responder for the signing authority certificate that validates the signing key each time that key is used in a signing operation. The OCSP Request shall indicate that the preferred signature algorithm to be used by the OCSP responder is RSA with SHA-256.

The SigningTime associated with the ATSC 3.0 application signature and the producedAt time of the corresponding OCSP Response providing the status of the signing authority certificate shall differ by no more than twenty-five (25) hours. The ATSC 3.0 application signing authority shall include the OCSP Response in the signed application and should not issue a signed application where the OCSP Response indicates that the status of the signing authority certificate (as specified in RFC 6960 [20]) is other than “good”.

The application signing authority shall include the object identifier id-ri-ocsp-response in the otherRevInfoFormat field and an OCSPResponse in the otherRevInfo field of each Cryptographic Message Syntax (RFC 5652 [13]) formatted digital signature contained in the signed multi-part MIME content. The OCSPResponse shall conform to the format specified in RFC 5940 [18].

A client uses the OCSP Response data that it receives to verify that the certificates that authenticate the application signing authority are valid at the time the application is signed. See CTA 2053 [28].

5.6 Pre-Shared Key Encrypted Connections

This section describes a general method by which two devices, known as the client device and the server device, can derive a pre-shared key and use that key to establish an encrypted connection. This method is based on the exchange of universally unique identifiers (UUID) [25] between the two devices and of the same input keying material (IKM) on each device. The derived pre-shared keys can then be used to establish a TLS 1.3 connection between the devices, using the TLS 1.3 Pre-Shared Key Exchange Parameters defined in Section 5.6.2.

Implementation of this section requires the implementation of all of the normative provisions of this Section 5.6.

When this section is used to establish an encrypted connection between a Companion Device (CD) application and a Primary Device (PD) per A/338 [29], the CD acts as the client and the PD acts as the server.

5.6.1 Pre-Shared Key Registration

5.6.1.1 Pre-Shared Key Identifier

Each pre-shared key installed on a client shall be referenced by the universally unique identifier (UUID) of the corresponding server with which it shares the key.

Each pre-shared key installed on a server shall be referenced by the UUID of the corresponding client with which it shares the key.

For example, UUIDs are provided in the device discovery protocol specified in A/338 [29].

5.6.1.2 Pre-Shared Key Hash Algorithm

The pre-shared key shall be used with the sha256 hash algorithm in the TLS 1.3 Key Schedule process (see Section 7.1 of [23]) when deriving secrets for use in TLS 1.3.

5.6.1.3 Pre-Shared Key Generation

The pre-shared key shall be derived from input keying material (IKM) using the PBKDF2 algorithm specified in RFC 8018 [22], as follows:

1) Create a salt by concatenating the server's 128-bit UUID and the client's 128-bit UUID in that order, giving a 256-bit binary value.
2) Set the pre-shared key to PBKDF2(IKM, salt, 50000, 32) using HMAC-sha256 as the underlying pseudorandom functions as described in RFC 8018 [22].

5.6.1.4 Key Generation Test Vectors

Correct implementation of the above pre-shared key generation using the below example input parameters yields the below output parameters.

```
Input:
Server UUID = 0x123e4567e89b12d3a456426655440000
Client UUID = 0x98734716276497582763764874687252
IKM = 'UserPassword' (0x5573657250617373776f7264)


Intermediate results:
Salt = 0x123e4567e89b12d3a45642665544000098734716276497582763764874687252


Output:
PSK = 0xf7a28206cfad1076eba1fce76245e012f357f5f70bcbe407f03d53ca8265de32
```

5.6.1.5 Initial Communication

When the pre-shared keys are derived, both client and server must be provided with IKM that consists of 32 or fewer ASCII characters. Such provision of IKM to the client and server is out of scope of this document; however it is expected that the end-user will provide a passcode, PIN or similar as IKM to both client and server. IKM shall not be stored in persistent memory in either client or server, and the client and server shall not reuse IKM.

5.6.1.6 Pre-Shared Key Storage

The client and server shall store each pre-shared key in a trusted keystore which limits key usage to those algorithms and applications used to establish a TLS connection. The ability to enter new pre-shared keys into the trusted keystore or to delete pre-shared keys from the trusted keystore shall be limited to a Privileged Application on the client and server. If a secure hardware-based

trusted keystore is available on the client or server device, this should be used to store the pre-shared keys.

5.6.2 TLS 1.3 Pre-Shared Key Exchange Parameters

A client device acting as a TLS Client and a server device acting as a TLS Server may establish a TLS 1.3 connection using pre-shared keys derived according to Section 5.6.1. The TLS 1.3 Server Connection Negotiation parameters defined in Section 5.1.1.2 shall be used with the pre-shared keys to establish this connection.

The TLS client handshake request indicates the use of the TLS 1.3 protocol and the TLS server shall not negotiate a downgrade to a previous version of TLS. The TLS client shall set the Pre-Shared Key Exchange Mode to psk_dhe_ek to enable an ephemeral ECDHE key to be established. The TLS client handshake request is not expected to include early data and the TLS server shall not accept any early data received from the client.

Server devices that have established a TLS 1.3 connection using pre-shared keys should support TLS Session Resumption (see Section 5.1.1.6) for those connections.

5.6.2.1 Pre-Shared Key Hash Algorithm

The pre-shared key shall be used with the sha256 hash algorithm in the TLS 1.3 Key Schedule process (see Section 7.1 of [19]) when deriving secrets for use in TLS 1.3.

5.7 Content Protection

5.7.1 Common Encryption

ATSC 3.0 uses the DASH-IF ATSC Profile [3] as the media container that will be sent through the broadcast emission to the receiver for consumption. MPEG Common Encryption (CENC) [2] has been specified as a digital rights management system suitable for use with ISO BMFF. Any media that requires DRM encryption shall use MPEG Common Encryption (CENC).

5.7.2 CENC and EME Support

ATSC 3.0 service and content may be protected using common encryption and one or more DRM systems. Multiple licenses to a single service or content may be available through multiple DRM systems simultaneously.

A DRM-protected ATSC 3.0 service or content shall be encrypted according to the Common Encryption standard [2] using the AES-128 algorithm in the CTR (‘cenc’), the CBC (‘cbc1’), or the CBCS (‘cbcs’) mode.

Annex A: ASN .1 Object Identifiers

A.1 ATSC REGISTERED OBJECT IDENTIFIERS

Table A.1 defines the ASN.1 Object Identifiers that are referenced in this document. Each of these identifiers is managed by ATSC under its IANA assigned Private Enterprise Number, which has the ASN.1 Object Identifier **1.3.6.1.4.1.51552** and is abbreviated to id-atsc in the table below.

Table A.1 ATSC Registered Object Identifiers

Identifier	Description	Prefix	Suffix
id-atsc-kp-author	ATSC Application Author Key Purpose	id-atsc	**.37.1**
id-atsc-kp-distributor	ATSC Application Distributor Key Purpose	id-atsc	**.37.2**
id-atsc-kp-signalingSigning	ATSC Broadcast Signaling Signing Key Purpose	id-atsc	**.37.3**
id-atsc-sdattr-bsid	ATSC Subject Directory Attribute for Broadcast Stream Identifier	id-atsc	**.9.1**

A.2 OTHER REFERENCED OBJECT IDENTIFIERS

Table A.2 defines the ASN.1 Object Identifiers referenced in this document, which are managed by IETF under the PKIX ASN.1 Object Identifier **1.3.6.1.5.5.7** (abbreviated to id-pkix in the table below).

Table A.2 Other Referenced Object Identifiers

Identifier	Description	Prefix	Suffix	Reference
id-kp	Key Purposes	id-pkix	**.3**	RFC 5280 [10]
id-kp-serverAuth	Server Authentication Key Purpose	id-kp	**.1**	RFC 5280 [10]
id-kp-codeSigning	Code Signing Key Purpose	id-kp	**.3**	RFC 5280 [10]
id-kp-OCSPSigning	OCSP Signing Key Purpose	id-kp	**.9**	RFC 6960 [21]
id-ri	Other Revocation Information	id-pkix	**.16**	RFC 5940 [18]
id-ri-ocsp-response	OCSP Response Revocation Information	id-ri	**.2**	RFC 5940 [18]

– End of Document –